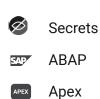
Code Smell 61





**c** C

C++

CloudFormation

COBOL COBOL

C# C#

g css

**⋈** Flex

**-co** Go

HTML

Kotlin

Kubernetes

Objective C

PHP

PL/I

PL/SQL PL/SQL

Python

RPG RPG

Ruby

Swift

Scala

Terraform

Text

TS TypeScript

T-SQL

VB VB.NET

VB6 VB6

XML XML



## Flex static code analysis

Unique rules to find Bugs, Security Hotspots, and Code Smells in your FLEX code

**∰** Bug

9

Security Hotspot (1)

ll rules				rability	
be used	y.allowD I in a tig	htly fo			
6 Vulr	nerability	/			
	stem.So y should				
<b>6</b> Vulr	nerability	/			
_	c classe	es sho	ould no	t be us	ed
Cod	e Smell				
configu domain	connecti red to n s with w r Flex ap	arrow vhich	ly spe local c	cify the onnecti	
<b>6</b> Vulr	nerability	/			
"default	" clause	es sho	uld be	first or	last
Cod	e Smell				
Event ty metada	pes sho ta tags	ould b	e defir	ned in	
Cod	e Smell				
	ames sł t listene		not be	hardco	ded
Cod	e Smell				
The spe	ecial "sta	ar" typ	e sho	uld not	be
Cod	e Smell				
Variable	es of the	e "Obje	ect" ty	pe shou	ıld
Cod	e Smell				
Method	ls shoul	d not	be em	pty	
Cod	e Smell				
	nt name conven		uld co	mply wi	ith a
_	e Smell				

All branches in a conditional structure should not have exactly the same

Classes that extend "Event" should

implementation

📆 Bug

Tags Search by name... "with" statements should not Analyze your code be used 🙀 Bug 🔷 Major 🕝 Never use with statements, since they decrease readability. When you do not specify a variable's scope, you do not always know where you are setting properties, so your code can be confusing. **Noncompliant Code Example** with (foo) { // Noncompliant return x; // is it a property of foo or local variabl **Compliant Solution** return foo.x; Available In: sonarcloud 🙆 | sonarqube

© 2008-2022 SonarSource S.A., Switzerland. All content is copyright protected. SONAR, SONARSOURCE, SONARLINT, SONARQUBE and SONARCLOUD are trademarks of SonarSource S.A. All other trademarks and copyrights are the property of their respective owners. All rights are expressly reserved.

Privacy Policy

override "Event.clone()"  • Bug
Constructors should not dispatch events
Rug
"ManagedEvents" tags should have companion "Event" tags
<b>∰</b> Bug
Objects should not be instantiated inside a loop
Two branches in a conditional structure should not have exactly the same implementation
Constructor bodies should be as lightweight as possible
Only "while", "do" and "for" statements should be labelled
Statements, operators and keywords specific to ActionScript 2 should not be used
"for" loop stop conditions should be invariant
Unused function parameters should be removed
Code Smell